**VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES**

**VIVEKANANDA SCHOOL OF INFORMATION TECHNOLOGY**



**BACHELOR OF COMPUTER APPLICATION**

**Practical –XII Java Lab**

**BCA 272**

**Guru Gobind Singh Indraprastha University   
Sector - 16C Dwarka, Delhi – 110078**



**SUBMITTED TO: SUBMITTED BY:**

Karthik Nair

**INDEX**

| S.No. | Practical | Page No. | Signature |
| --- | --- | --- | --- |
| 1. | Write a Java program to print all odd numbers between 1 to 10. | 4 |  |
| 2. | Write a Java program to find out factorial of a number through recursion. | 5 |  |
| 3. | Write a Java program to accept command line arguments & print them. | 6 |  |
| 4. | Write a Java program to print Fibonacci series. | 7 |  |
| 5. | Write a Java program that creates a class accounts with following details:  Instance variables: ac\_no., name, ac\_name, balance  Methods: withdrawal (), deposit (), display ().use constructors to initialize members. | 8 |  |
| 6. | Write a Java program to implement constructor overloading. | 10 |  |
| 7. | Write a Java program to count the no. of objects created in a program. | 11 |  |
| 8. | Write a Java program to show call by value & call by reference. | 12 |  |
| 9. | Write a Java program to implement method overriding & method overloading. | 14 |  |
| 10. | Create a class box having height, width, depth as the instance variables & calculate its volume. Implement constructor overloading in it. Create a subclass named box\_new that has weight as an instance variable. Use super in the box\_new class to initialize members of the base class | 15 |  |
| 11. | Write a Java program to implement run time polymorphism | 17 |  |
| 12. | Write a Java program to implement interface. Create an interface named shape having area () & perimeter () as its methods. Create three classes circle, rectangle & square that implement this interface. | 18 |  |
| 13. | Write a Java program to show multiple inheritance. |  |  |
| 14. | Write a Java program to implement exception handling. Use try, catch & finally |  |  |
| 15. | Write a Java program to implement matrix multiplication by 2d array |  |  |
| 16. | Write a Java program to implement vector [use: addelement(),elementat().removeelement(),size().] |  |  |
| 17. | Create a user defined exception named “nomatchexception” that is fired when the string entered by the user is not “india” |  |  |
| 18. | Write a Java program to show even & odd numbers by thread |  |  |
| 21. | Write a Java program to demonstrate the use of equals(), trim() ,length() , substring(), compareto() of string class. |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| 22. | Write a Java program to implement vector.  [use: addelement(),elementat().removeelement(),size().] |  |  |
| 23. | Write a Java program that draws different color shapes on an applet .set the foreground & background color as red & blue. |  |  |
| 24. | Write a Java program to show moving banner by applet. |  |  |
| 25. | Write a Java program to demonstrate the use of equals() and == in Java. |  |  |
| 26. | Write a Java program to implement all mouse events and mouse motion events. |  |  |
| 27. | Write a Java program to implement keyboard events |  |  |
| 28. | Write a Java program using AWT to create a simple calculator. |  |  |
| 29. | Create a login form using AWT controls like labels, buttons, textboxes, checkboxes, list, checkboxgroup. The selected checkbox item names should be displayed |  |  |
| 30. | Create a login form using Swing controls like Jlabels, Jbuttons, Jtextboxes, Jcheckboxes. |  |  |
| 31. | Write a Java program to show all layout managers. (4 layout managers) |  |  |
| 32. | Create an applet with two buttons named ‘*audio’* and ‘*image’*. |  |  |
| 33. | Create a Java applet with three buttons ‘red’,’green’,’blue’. Whenever user presses any button the corresponding color should be seen as background color in an applet window. |  |  |
| 34. | Write a Java program in Java to implement the concept of ‘synchronization’ using thread |  |  |
| 35. | Create a simple JDBC program that creates a table, stores data into it, retrieves & prints the data. |  |  |
| 36. | Write a Java program in Java to create database table using Java. |  |  |
| 37. | Write a Java program in Java to insert, update, delete & select records. |  |  |
| 38. | Write Java program to read input from java console |  |  |
| 39. | Write a Java program to implement file handling(both reading & writing to a file) |  |  |
| 40. | Write a Java program on anonymous classes |  |  |

**1. Write a Java program to print all odd numbers between 1 to 10.**

**Code:**

// This code prints odd numbers from 1 to n

// where n is the input for the program

public class aPrintOddNumbers{

public static void main(String[] args) {

int n = 10;

for (int i = 1; i <= n; i+=2) {

System.out.print(i+" ");

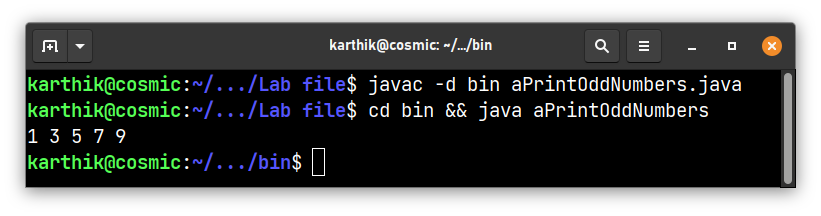
}

System.out.println();

}

}

**Output:**

****

**2. Write a Java program to find out factorial of a number through recursion**

**Code:**

// This is a recursive function which finds the factorial of an integer n.

// n is the input integer.

// The function returns the factorial of n.

class bFactorialThroughRecursion{

static int factorial(int n){

if (n == 0)

return 1; // base case

return n\*factorial(n-1); // recursive call

}

public static void main(String[] args){

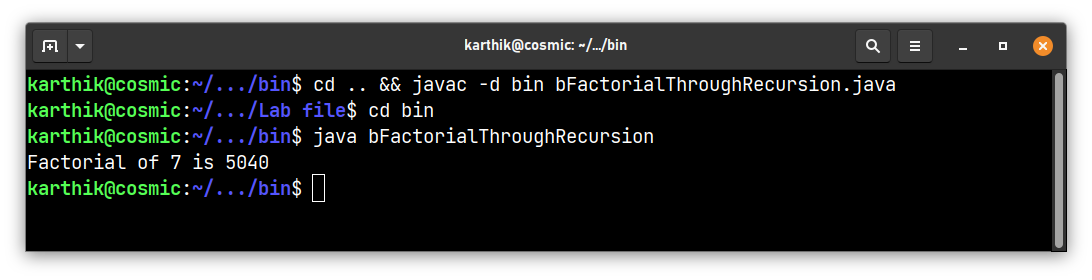
int n = 7;

System.out.println("Factorial of " + n + " is " + factorial(n));

}

}

**Output:**



**3. Write a Java program to accept command line arguments & print them**

**Code:**

// This code displays the contents of an array of strings passed as command line argument to the main method

// The array is iterated through and the contents are displayed, one per line

class cCommandLineArgs{

public static void main(String[] args){

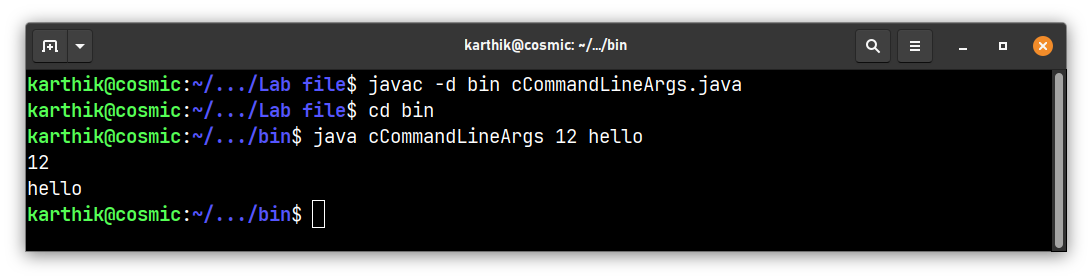
for (int i=0; i<args.length; i++)

System.out.println(args[i]);

}

}

**Output:**

****

**4. Write a Java program to print Fibonacci series**

**Code:**

class dFibonacci{

// This function prints the fibonacci series

// n is the input for the function

static void fibonacci(int n){

int a = 0, b = 1, c;

System.out.print(a+" "+b+" ");

for (int i=2; i<n; i++){

c = a + b; // next term

System.out.print(c+" ");

a = b; // update a

b = c; // update b

}

System.out.println();

}

public static void main(String[] args) {

int n = 10;

if (n > 0)

fibonacci(n);

else if (n == 0) // exceptional case

System.out.println("0");

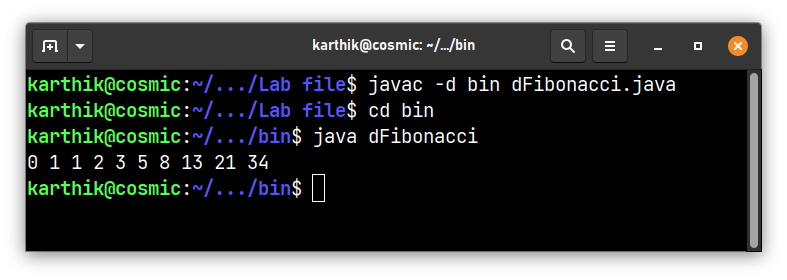
else // exceptional case

System.out.println("Invalid input");

}

}

**Output:**



**5. Write a Java program that creates a class accounts with following details:**

**Instance variables: ac\_no., name, ac\_name, balance**

**Methods: withdrawal (), deposit (), display ().use constructors to initialize members**

**Code:**

class eAccounts{

// instance variables

int ac\_no;

String name, ac\_name;

double balance;

// constructor

eAccounts(int ac\_no, String name, String ac\_name, double balance){

this.ac\_no = ac\_no;

this.name = name;

this.ac\_name = ac\_name;

this.balance = balance;

}

// this method withdraws amount from balance

void withdrawal(double amount){

if (amount > balance)

System.out.println("Insufficient balance");

else

balance -= amount;

}

// this method deposits amount to balance

void deposit(double amount){

balance += amount;

}

// this method displays account details

void display(){

System.out.println("Account number: " + ac\_no);

System.out.println("Name: " + name);

System.out.println("Account name: " + ac\_name);

System.out.println("Balance: " + balance+"\n");

}

public static void main(String[] args) {

// create object of eAccounts class

eAccounts e = new eAccounts(123456, "John", "Savings", 10000);

// before withdrawal

System.out.println("Before withdrawal:");

e.display();

// withdraw 5000

e.withdrawal(5000);

// after withdrawal

System.out.println("After withdrawal of 5000:");

e.display();

// deposit 1000

e.deposit(1000);

// after deposit

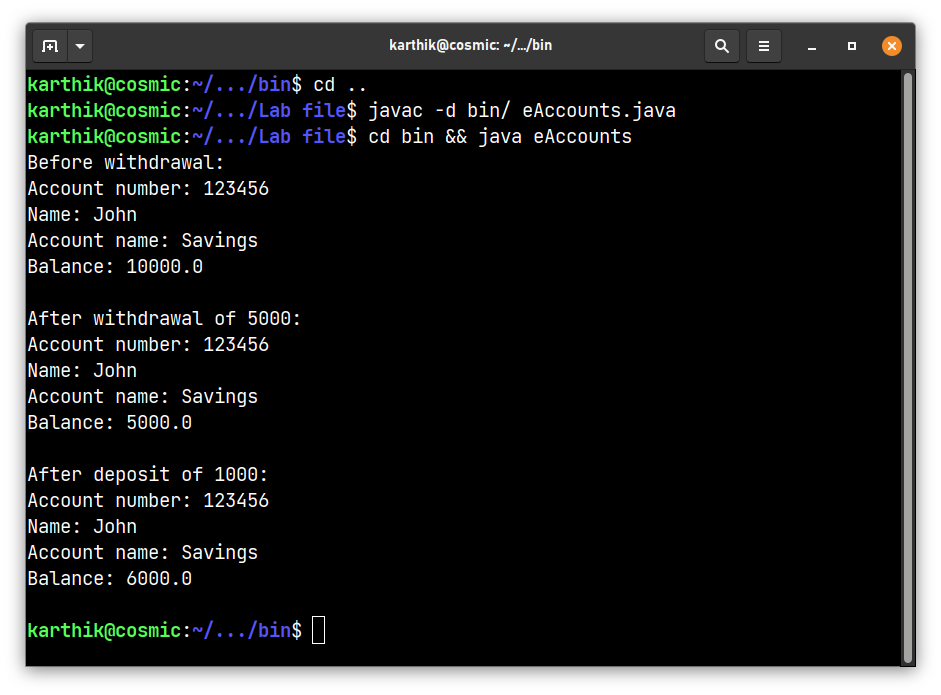
System.out.println("After deposit of 1000:");

e.display();

}

}

**Output:**



**6. Write a Java program to implement constructor overloading**

**Code:**

class fConstructorOverloading{

int a, b;

// This is the default constructor

fConstructorOverloading(){

a = 0;

b = 0;

}

// This is the parameterized constructor

fConstructorOverloading(int x, int y){

a = x;

b = y;

}

// This is the copy constructor

fConstructorOverloading(fConstructorOverloading obj){

a = obj.a;

b = obj.b;

}

// This function prints the values of a and b

void print(){

System.out.println("a = " + a + " b = " + b);

}

public static void main(String[] args) {

fConstructorOverloading obj1 = new fConstructorOverloading();

fConstructorOverloading obj2 = new fConstructorOverloading(10, 20);

fConstructorOverloading obj3 = new fConstructorOverloading(obj2);

obj1.print();

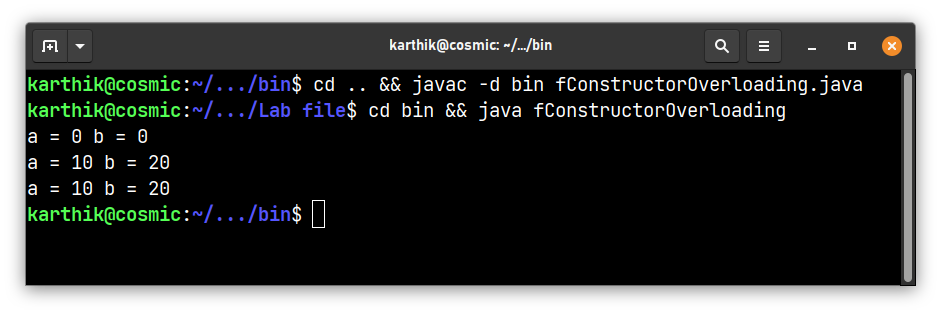
obj2.print();

obj3.print();

}

}

**Output:**



**7. Write a Java program to count the no. of objects created in a program**

**Code:**

class CountObjects{

static int count = 0;

CountObjects(){

// this function increments the count of objects created

count++;

}

static int getCount(){

// this function returns the count of objects created

return count;

}

}

class gObjectCounter{

public static void main(String[] args)

{

// create two objects of CountObjects class

CountObjects var1= new CountObjects();

CountObjects var2= new CountObjects();

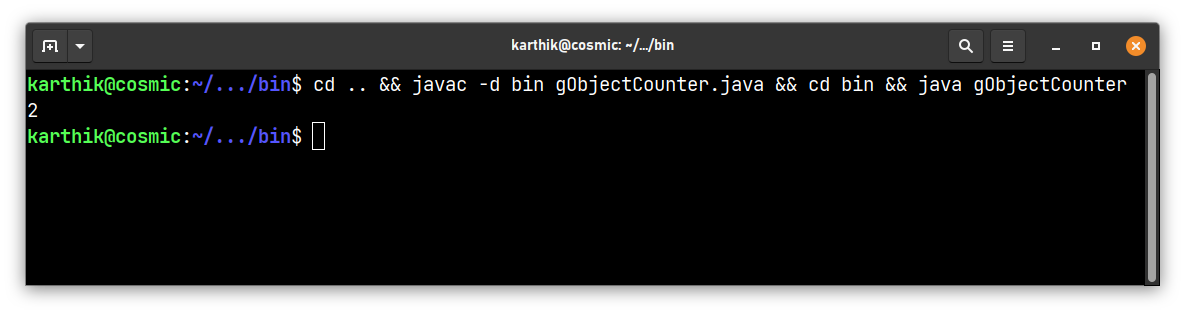
// print the count of objects created using getCount() function

System.out.println(CountObjects.getCount());

}

}

**Output:**



**8. Write a Java program to show call by value & call by reference**

**Output:**

class CalcByValue{

// this function increments the value of a by 1 using value

void increment(int a){

a++;

}

}

class CalcByReference{

// this function increments the value of a by 1 using reference

int a;

CalcByReference(int i){

a = i;

}

void increment(CalcByReference obj){

obj.a++;

}

}

class hCallByValueReference{

public static void main(String[] args) {

// call by value

int a = 10;

System.out.println("Call by value");

CalcByValue obj1 = new CalcByValue();

System.out.println("Before incrementing a = " + a);

obj1.increment(a);

System.out.println("After incrementing a = " + a);

// call by reference

System.out.println("Call by reference");

CalcByReference obj2 = new CalcByReference(10);

System.out.println("Before incrementing a = " + obj2.a);

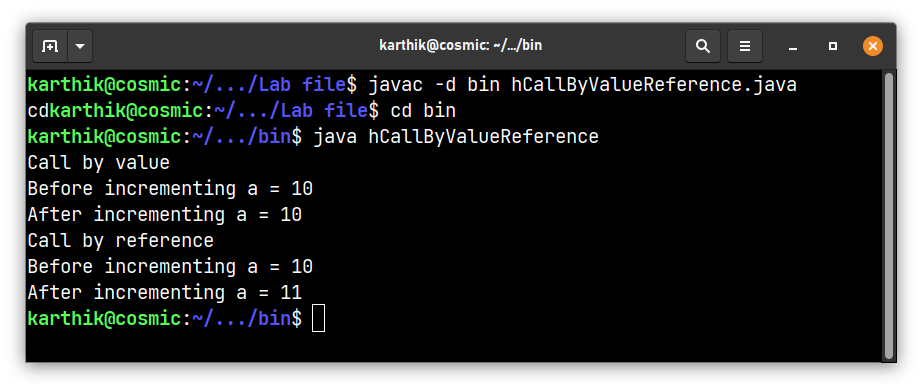
obj2.increment(obj2); // passing the reference of obj2 to the function

System.out.println("After incrementing a = " + obj2.a);

}

}

**Output:**

****

**9. Write a Java program to implement method overriding & method overloading.**

class Book{

void info(){

System.out.println("This is a book");

}

}

class Novel extends Book{

// method overriding

void info(){

System.out.println("This is a novel");

}

// method overloading

void info(int x){

System.out.println("This is a novel with " + x + " chapters");

}

}

class fMethodOverridingOverloading{

public static void main(String[] args) {

// parent class method is called

Book obj1 = new Book();

obj1.info();

// child class method is called

// method overriding

Novel obj2 = new Novel();

obj2.info();

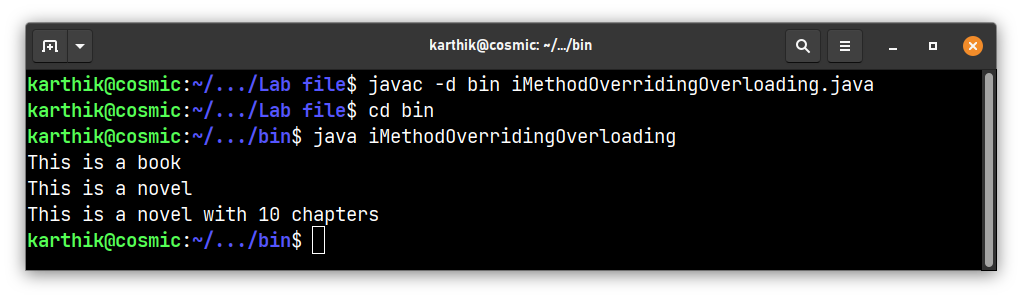
// method overloading

obj2.info(10);

}

}

**Output:**



**10. Create a class box having height, width, depth as the instance variables & calculate its volume. Implement constructor overloading in it. Create a subclass named box\_new that has weight as an instance variable. Use super in the box\_new class to initialize members of the base class**

**Code:**

class box{

int height, width, depth;

// constructor overloading

box(){

height = 0;

width = 0;

depth = 0;

}

box(int height, int width, int depth){

this.height = height;

this.width = width;

this.depth = depth;

}

box(box obj){

height = obj.height;

width = obj.width;

depth = obj.depth;

}

int volume(){

return height\*width\*depth;

}

}

class box\_new extends box{

// instance variable

int weight;

box\_new(){

super();

weight = 0;

}

box\_new(int height, int width, int depth, int weight){

super(height, width, depth);

this.weight = weight;

}

box\_new(box\_new obj){

super(obj);

weight = obj.weight;

}

void print(){

System.out.println("Height: " + height);

System.out.println("Width: " + width);

System.out.println("Depth: " + depth);

System.out.println("Weight: " + weight);

System.out.println("Volume: " + volume());

}

}

class jqn10{

public static void main(String[] args) {

box\_new obj1 = new box\_new();

box\_new obj2 = new box\_new(10, 20, 30, 40);

box\_new obj3 = new box\_new(obj2);

obj1.print();

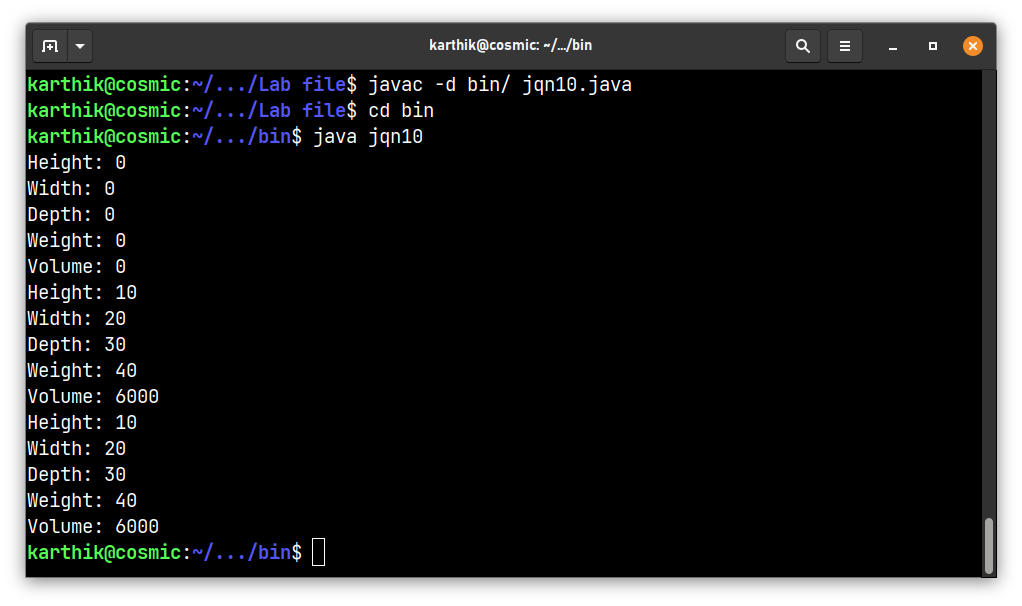
obj2.print();

obj3.print();

}

}

**Output:**



**11. Write a Java program to implement run time polymorphism**

**Code:**

class Book{

void info(){

System.out.println("This is a book");

}

}

class Novel extends Book{

// info is overridden

void info(){

System.out.println("This is a novel");

}

}

class kRunTimePolymorphism{

public static void main(String[] args) {

Book book0= new Novel();

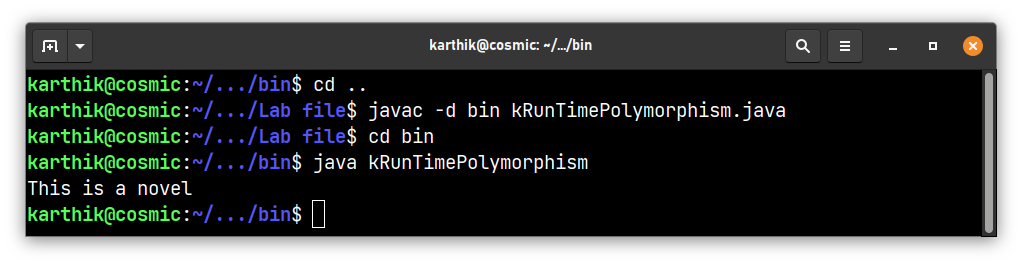
// calling the run method by the reference variable of the parent class

book0.info();

}

}

**Output:**



**12. Write a Java program to implement interface. Create an interface named shape having area () & perimeter () as its methods. Create three classes circle, rectangle & square that implement this interface**

**Code:**

interface shape{

// abstract methods

void area();

void perimeter();

}

// circle, rectangle and square classes implement the shape interface

class circle implements shape{

int r;

circle(int r){

this.r = r;

}

public void area(){

System.out.println("Area of circle is: "+(3.14\*r\*r));

}

public void perimeter(){

System.out.println("Perimeter of circle is: "+(2\*3.14\*r));

}

}

class rectangle implements shape{

int l,b;

rectangle(int l,int b){

this.l = l;

this.b = b;

}

public void area(){

System.out.println("Area of rectangle is: "+(l\*b));

}

public void perimeter(){

System.out.println("Perimeter of rectangle is: "+(2\*(l+b)));

}

}

class square implements shape{

int s;

square(int s){

this.s = s;

}

public void area(){

System.out.println("Area of square is: "+(s\*s));

}

public void perimeter(){

System.out.println("Perimeter of square is: "+(4\*s));

}

}

class lInterfaceImplementation{

public static void main(String[] args) {

// create objects of circle, rectangle and square and call their methods

circle c = new circle(5);

c.area();

c.perimeter();

rectangle r = new rectangle(5,6);

r.area();

r.perimeter();

square s = new square(5);

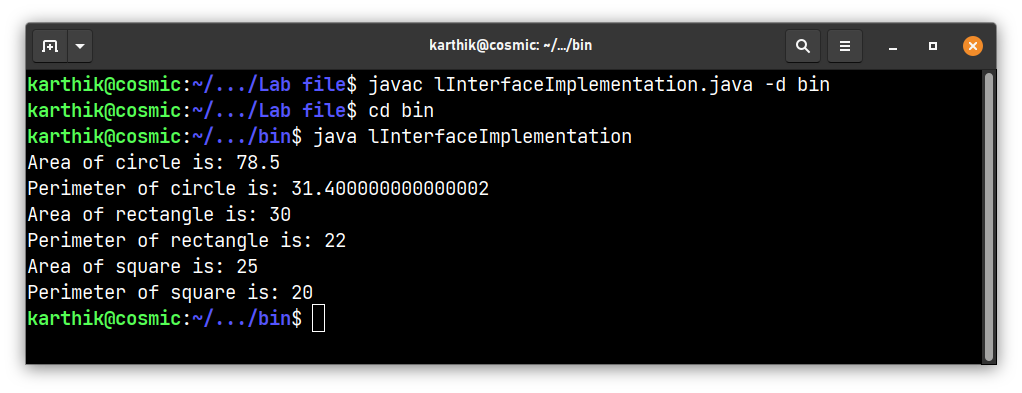
s.area();

s.perimeter();

}

}

**Output:**



**13. Write a Java program to show multiple inheritance**

**Code:**

// in java, multiple inheritance is not supported

// but it can be achieved using interfaces

interface Animal{

// abstract methods

public void eat();

public void move();

}

interface Mammal{

// abstract method

public void giveBirth();

}

class Dog implements Animal, Mammal{

// multiple inheritance is achieved using interfaces

public void eat(){

System.out.println("The dog is eating.");

}

public void move(){

System.out.println("The dog is moving.");

}

public void giveBirth(){

System.out.println("The dog is giving birth.");

}

}

public class mMultipleInheritance {

public static void main(String[] args){

Dog dog0 = new Dog();

dog0.eat();

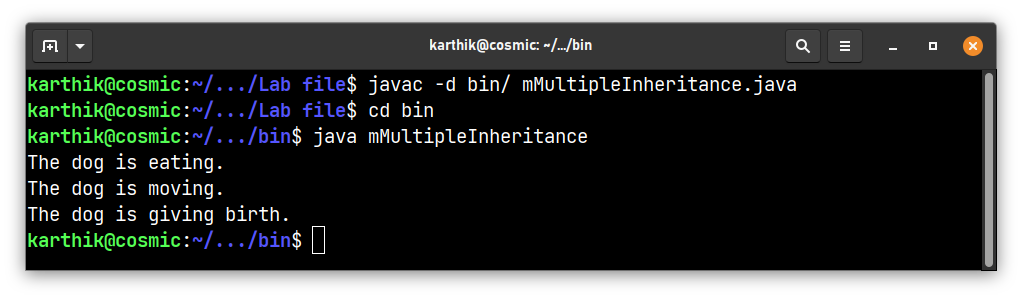
dog0.move();

dog0.giveBirth();

}

}

**Output:**



**14. Write a Java program to implement exception handling. Use try, catch & finally.**

**Code:**

public class nTryCatchFinally{

public static void main(String[] args) {

int a = 10;

int b = 0;

try{

// runs the code that may throw an exception

int c = a/b;

System.out.println("The result is: "+c);

}

catch(ArithmeticException e){

// handles the exception

System.out.println("Arithmetic Exception occured");

}

finally{

// executes the code regardless of the exception

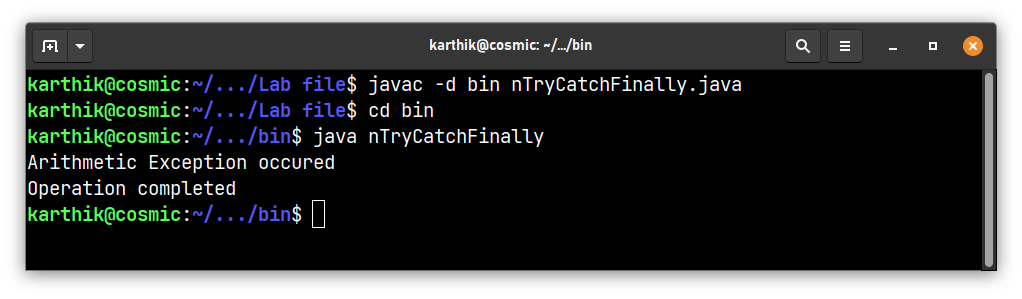
System.out.println("Operation completed");

}

}

}

**Output:**



**15. Write a Java program to implement matrix multiplication by 2d array**

**Code:**

public class oMatrixMul {

public static void main(String[] args) {

// create 2d arrays and multiply them

int a[][] = {{1,2,3},{4,5,6},{7,8,9}};

int b[][] = {{1,2,3},{4,5,6},{7,8,9}};

int c[][] = new int[3][3];

for(int i=0;i<3;i++){

for(int j=0;j<3;j++){

// multiply the matrices

c[i][j] = 0;

for(int k=0;k<3;k++){

// add the products of the elements of the matrices

c[i][j] += a[i][k]\*b[k][j];

}

}

}

// print the result

for(int i=0;i<3;i++){

for(int j=0;j<3;j++){

System.out.print(c[i][j]+" ");

}

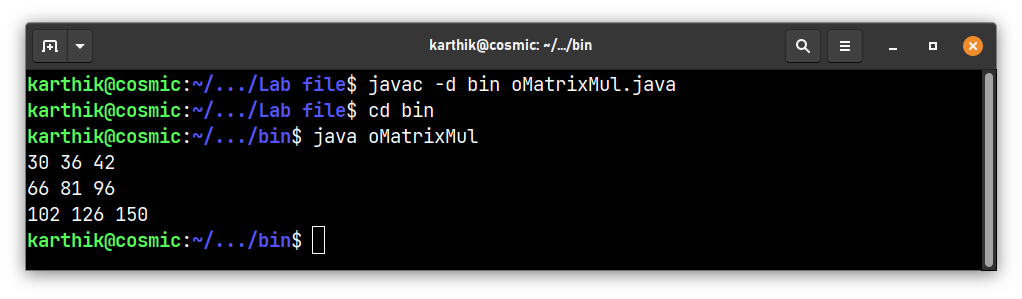
System.out.println();

}

}

}

**Output:**



**17. Create a user defined exception named “nomatchexception” that is fired when the string entered by the user is not “india”**

**Code:**

// import the required packages and classes to use the Scanner class

import java.util.Scanner;

// create a user defined exception

class nomatchexception extends Exception{

nomatchexception(String s){

super(s);

}

}

public class pUserDefinedException{

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter a string: ");

// read the string from the user

String s = sc.nextLine();

sc.close();

try{

// check if the string entered is india

if(s.equals("india")){

System.out.println("The string entered is: "+s);

}

else{

// throw the exception if the string is not india

throw new nomatchexception("The string entered is not india");

}

}

// catch the exception and print the message

catch(nomatchexception e){

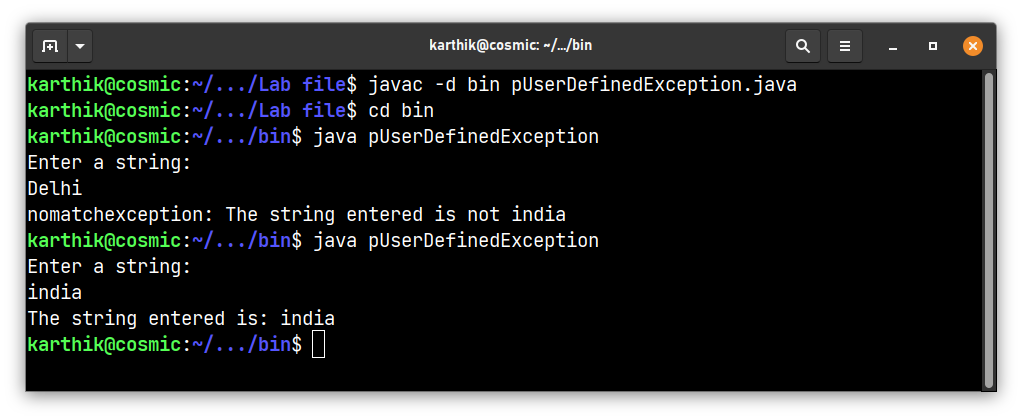
System.out.println(e);

}

}

}

**Output:**



**18. Write a Java program to show even & odd numbers by thread**

**Code:**

class Even extends Thread{

// initialize n

int n;

Even(int n){

this.n = n;

}

// override run method

public void run(){

for (int i=0; i<=n; i+=2){

System.out.print(i+" ");

}

System.out.println();

}

}

class Odd extends Thread{

// initialize n

int n;

Odd(int n){

this.n = n;

}

// override run method

public void run(){

for (int i=1; i<=n; i+=2){

System.out.print(i+" ");

}

System.out.println();

}

}

class rEvenOddThread{

public static void main(String[] args) {

// create object of Even class

Even e = new Even(10);

// create object of Odd class

Odd o = new Odd(10);

// start thread

e.start();

o.start();

}

}

**Output:**



**21. Write a Java program to demonstrate the use of equals(), trim() ,length() , substring(), compareto() of string class**

**Code:**

public class uQn21 {

public static void main(String[] args) {

String str1 = " Some Text ";

String str2 = "some text";

String str3 = "Some text";

String str4 = "text";

// equals() method returns true if the string is equal to the given string

System.out.println(str1.equals(str2)); // false

System.out.println(str2.equals(str3)); // true

// trim() method removes the leading and trailing spaces

System.out.println(str1.trim()); // "hello world"

// length() method returns the length of the string

System.out.println(str2.length()); // 11

// substring() method returns the substring of the string

System.out.println(str2.substring(6));

// compareTo() method returns the difference of the ascii values of the first unmatched character

System.out.println(str2.compareTo(str3));

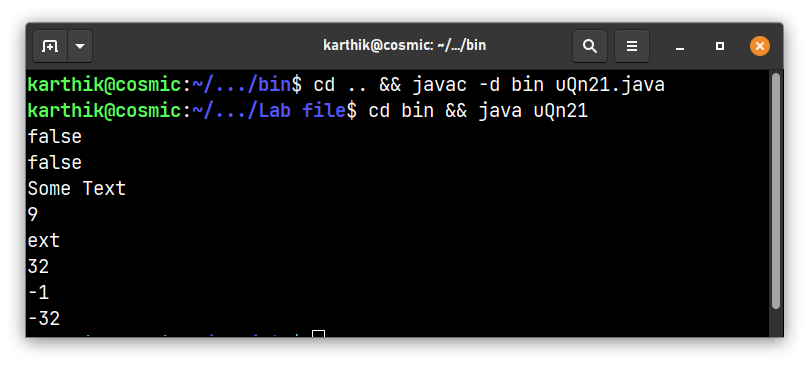
System.out.println(str2.compareTo(str4));

System.out.println(str3.compareTo(str2));

}

}

**Output:**



**22. Write a Java program to implement vector [use: delement(),elementat().removeelement(),size().]**

**Code:**

// Write a Java program to implement vector

// [use: addelement(),elementat().removeelement(),size().]

import java.util.Vector;

public class pVectorImplementation {

public static void main(String[] args){

Vector<Integer> v = new Vector<Integer>();

v.addElement(1);

v.addElement(2);

v.addElement(3);

v.addElement(4);

System.out.println("The vector is: "+v);

System.out.println("The size of the vector is: "+v.size());

System.out.println("The element at index 2 is: "+v.elementAt(2));

System.out.println("Removing element at index 2...");

v.removeElementAt(2);

System.out.println("The vector is: "+v);

System.out.println("The size of the vector is: "+v.size());

}

}

**Output:**



**23. Write a Java program that draws different color shapes on an applet .set the foreground & background color as red & blue.**

**Code:**

import java.awt.Color;

import java.awt.Graphics;

import javax.swing.\*;

public class sColorShapes extends JPanel {

public static void main(String[] args) {

JFrame frame = new JFrame("Color Shapes");

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setSize(400, 400);

frame.add(new sColorShapes());

frame.setVisible(true);

}

protected void paintComponent(Graphics g) {

super.paintComponent(g);

g.setColor(Color.RED); // Set the foreground color to red

g.fillRect(50, 50, 100, 100); // Draw a rectangle

// Draw a circle with 200, 50 as the center and 100 as the radius

g.fillOval(200, 50, 100, 100);

g.setColor(Color.BLUE);

int[] xPoints = {300, 350, 250}; // Draw a triangle

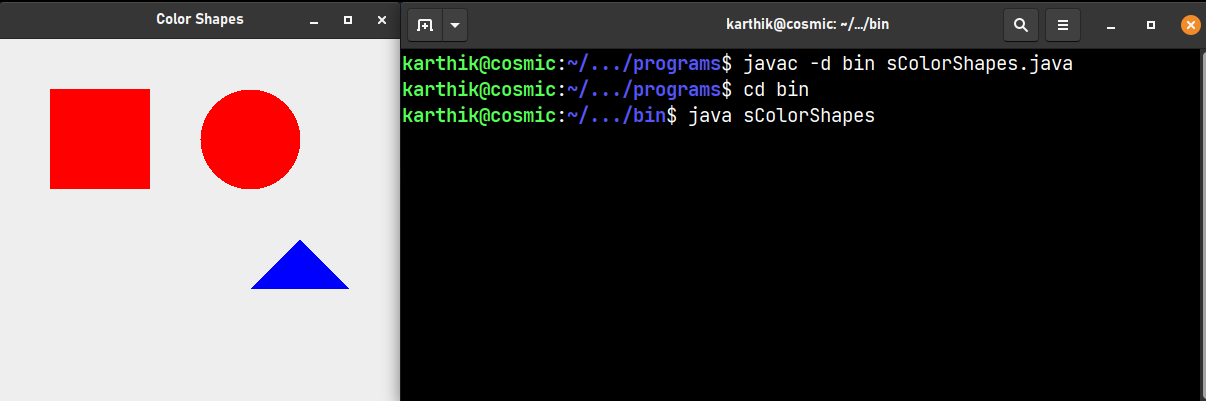
int[] yPoints = {200, 250, 250};

g.fillPolygon(xPoints, yPoints, 3);

}

}

**Output:**



**24. Write a Java program to show moving banner by applet**

**Code:**

import javax.swing.\*;

import java.awt.\*;

public class tMovingBanner extends JPanel implements Runnable {

private String message = "Moving Banner Text"; // The message to display

private int x = 0; // coordinates of the message

private int y = 100;

private int delay = 50; // The delay between updates

public tMovingBanner() {

Thread thread = new Thread(this); // updates the position of the banner

thread.start();

}

protected void paintComponent(Graphics g) {

super.paintComponent(g); // clears the screen and prepares it for redrawing

g.drawString(message, x, y); // draws the message at the specified coordinates

}

// Update the position of the banner.

// Override the run method of the Runnable interface.

public void run() {

while (true) {

x += 5;

if (x > getWidth()) {

x = -100;

}

repaint();

try {

Thread.sleep(delay);

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

public static void main(String[] args) {

JFrame frame = new JFrame("Moving Banner");

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setSize(400, 200);

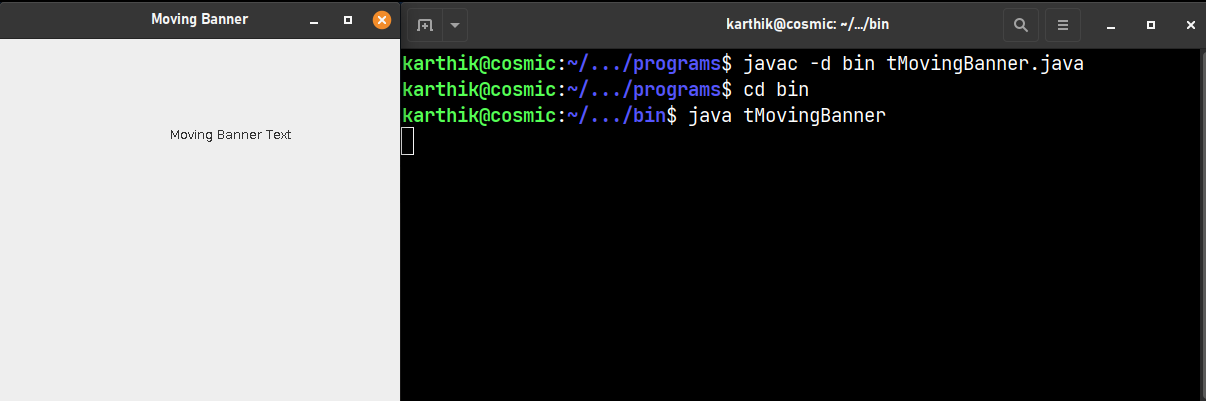
frame.add(new tMovingBanner());

frame.setVisible(true);

}

}

**Output:**



**25. Write a Java program to demonstrate the use of equals() and == in Java.**

**Code:**

public class vEquals {

public static void main(String[] args) {

String s1 = "Hello";

String s2 = "Hello";

String s3 = new String("Hello");

System.out.println("s1 == s2: " + (s1 == s2));

System.out.println("s1 == s3: " + (s1 == s3));

System.out.println("s1.equals(s2): " + s1.equals(s2));

System.out.println("s1.equals(s3): " + s1.equals(s3));

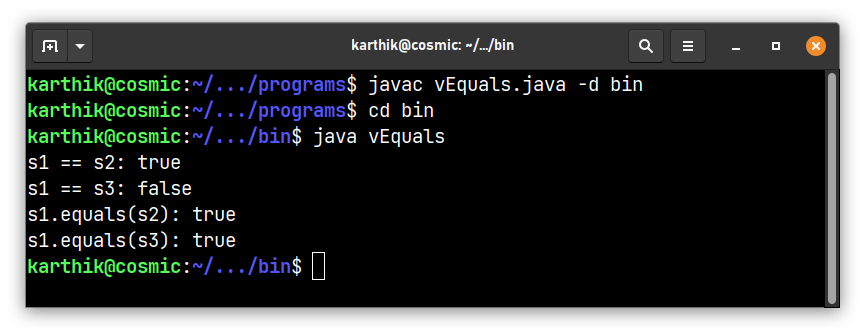
}

}

// equals method compares the contents of the string

// == compares the the memory location of the string

**Output:**



**26. Write a Java program to implement all mouse events and mouse motion events.**

**Code:**

import java.awt.\*;

import java.awt.event.\*;

public class wMouseMotionEvents extends Frame implements MouseListener, MouseMotionListener {

private Label label;

public wMouseMotionEvents() {

label = new Label();

add(label, BorderLayout.SOUTH); // add label to the bottom of the frame

addMouseListener(this);

addMouseMotionListener(this);

setSize(300, 200);

setVisible(true);

}

public static void main(String[] args) {

new wMouseMotionEvents();

}

public void mouseClicked(MouseEvent e) {

label.setText("Mouse clicked at (" + e.getX() + ", " + e.getY() + ")");

}

public void mousePressed(MouseEvent e) {

label.setText("Mouse pressed at (" + e.getX() + ", " + e.getY() + ")");

}

public void mouseReleased(MouseEvent e) {

label.setText("Mouse released at (" + e.getX() + ", " + e.getY() + ")");

}

public void mouseEntered(MouseEvent e) {

label.setText("Mouse entered at (" + e.getX() + ", " + e.getY() + ")");

}

public void mouseExited(MouseEvent e) {

label.setText("Mouse exited at (" + e.getX() + ", " + e.getY() + ")");

}

public void mouseDragged(MouseEvent e) {

label.setText("Mouse dragged at (" + e.getX() + ", " + e.getY() + ")");

}

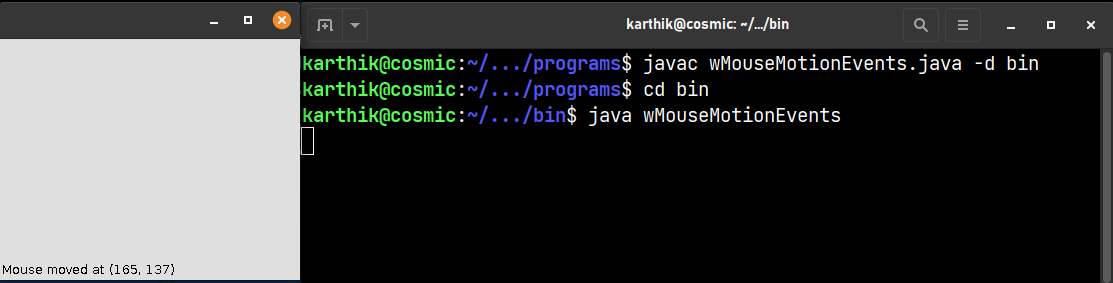
public void mouseMoved(MouseEvent e) {

label.setText("Mouse moved at (" + e.getX() + ", " + e.getY() + ")");

}

}

**Output:**



**27. Write a Java program to implement keyboard events**

**Code:**

import java.awt.\*;

import java.awt.event.\*;

public class xKeyboardEvents extends Frame implements KeyListener {

private Label label;

public xKeyboardEvents() {

label = new Label();

add(label, BorderLayout.SOUTH);

addKeyListener(this);

setSize(300, 200);

setVisible(true);

}

public static void main(String[] args) {

new xKeyboardEvents();

}

public void keyPressed(KeyEvent e) {

label.setText("Key pressed: " + e.getKeyChar());

}

public void keyReleased(KeyEvent e) {

label.setText("Key released: " + e.getKeyChar());

}

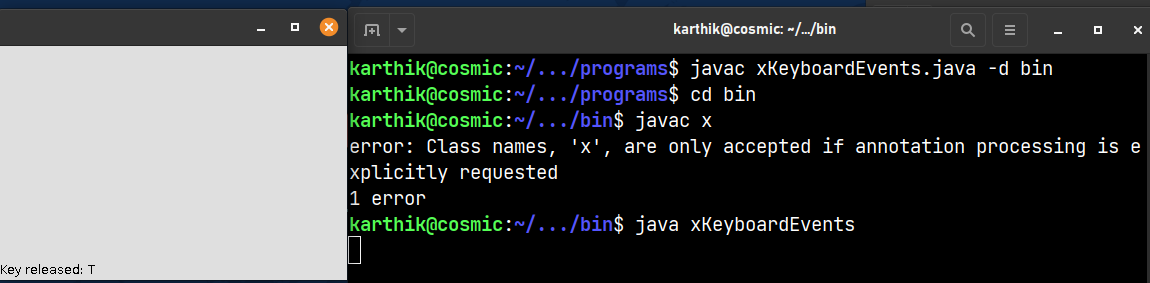
public void keyTyped(KeyEvent e) {

label.setText("Key typed: " + e.getKeyChar());

}

}

**Output:**



**28. Write a Java program using AWT to create a simple calculator**

**Code:**

import java.awt.\*;

import java.awt.event.\*;

public class ySimpleCalc extends Frame implements ActionListener {

private TextField num1, num2, result;

private Button add, subtract, multiply, divide;

public ySimpleCalc() {

num1 = new TextField(10);

num2 = new TextField(10);

result = new TextField(10);

result.setEditable(false);

add = new Button("+");

subtract = new Button("-");

multiply = new Button("\*");

divide = new Button("/");

add.addActionListener(this);

subtract.addActionListener(this);

multiply.addActionListener(this);

divide.addActionListener(this);

Label label1 = new Label("Number 1:");

Label label2 = new Label("Number 2:");

Label label3 = new Label("Result:");

setLayout(new GridLayout(3, 4));

add(label1);

add(num1);

add(label2);

add(num2);

add(add);

add(subtract);

add(multiply);

add(divide);

add(label3);

add(result);

setTitle("Simple Calculator");

setSize(250, 200);

setVisible(true);

}

public static void main(String[] args) {

new ySimpleCalc();

}

public void actionPerformed(ActionEvent e) {

int n1 = Integer.parseInt(num1.getText());

int n2 = Integer.parseInt(num2.getText());

int r = 0;

if (e.getSource() == add) {

r = n1 + n2;

} else if (e.getSource() == subtract) {

r = n1 - n2;

} else if (e.getSource() == multiply) {

r = n1 \* n2;

} else if (e.getSource() == divide) {

r = n1 / n2;

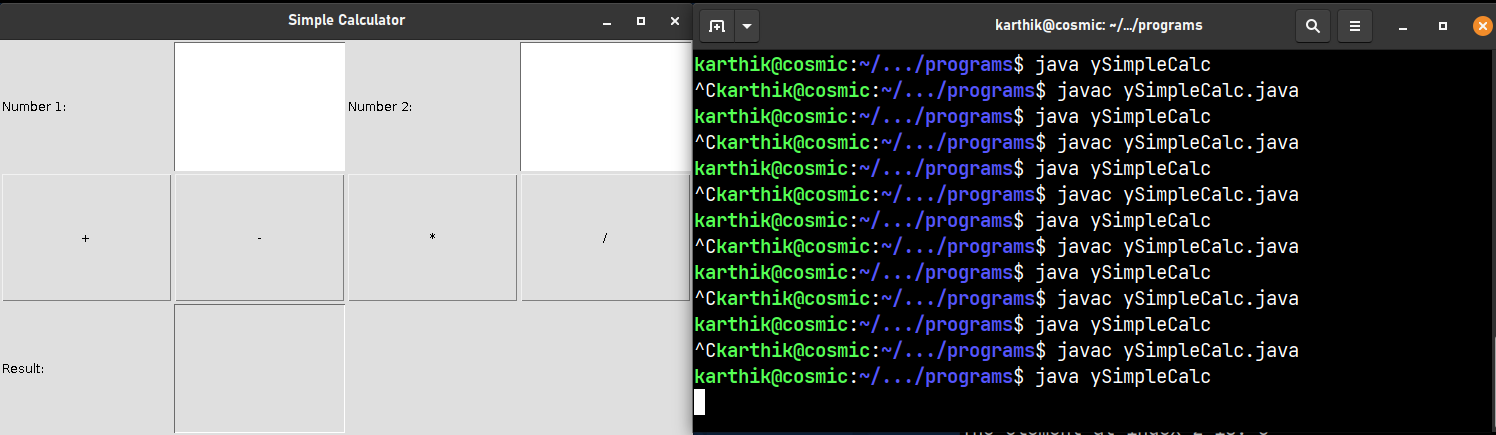
}

result.setText(String.valueOf(r));

}

}

**Output:**



**29. Create a login form using AWT controls like labels, buttons, textboxes, checkboxes, list, checkboxgroup. The selected checkbox item names should be displayed.**

**Code:**

import java.awt.\*;

import java.awt.event.\*;

public class zLoginForm extends Frame implements ActionListener {

private TextField usernameField;

private TextField passwordField;

private Checkbox rememberMeCheckbox;

private Button loginButton;

public zLoginForm() {

// frame

setTitle("Login Form");

setSize(300, 200);

addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent e) {

dispose();

}

});

// create the components

Label usernameLabel = new Label("Username:");

Label passwordLabel = new Label("Password:");

usernameField = new TextField(20);

passwordField = new TextField(20);

passwordField.setEchoChar('\*');

rememberMeCheckbox = new Checkbox("Remember me");

loginButton = new Button("Login");

loginButton.addActionListener(this);

// set up the layout

setLayout(new GridLayout(4, 2));

add(usernameLabel);

add(usernameField);

add(passwordLabel);

add(passwordField);

add(new Label());

add(rememberMeCheckbox);

add(new Label());

add(loginButton);

setVisible(true);

}

public static void main(String[] args) {

new zLoginForm();

}

public void actionPerformed(ActionEvent e) {

String username = usernameField.getText();

String password = passwordField.getText();

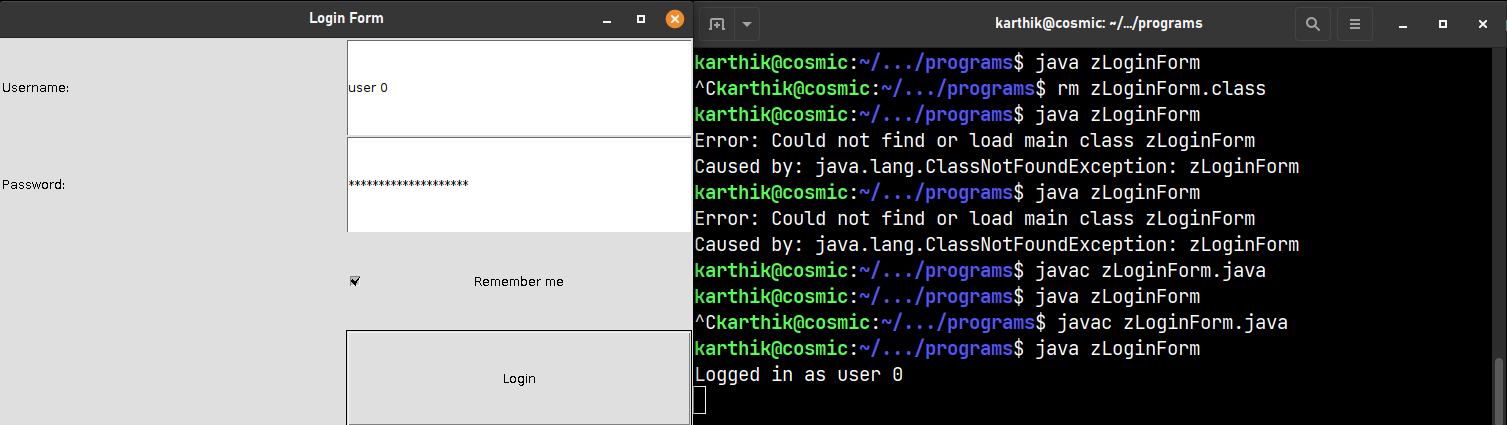
boolean rememberMe = rememberMeCheckbox.getState();

System.out.println("Logged in as " + username);

}

}

**Output:**

****

**30. Create a login form using Swing controls like Jlabels, Jbuttons, Jtextboxes, Jcheckboxes.**

**Code:**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class aLoginFormSwing extends JFrame implements ActionListener {

private JTextField usernameField;

private JPasswordField passwordField;

private JCheckBox rememberMeCheckbox;

private JButton loginButton;

public aLoginFormSwing() {

// frame

setTitle("Login Form");

setSize(300, 200);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

// components

JLabel usernameLabel = new JLabel("Username:");

JLabel passwordLabel = new JLabel("Password:");

usernameField = new JTextField(20);

passwordField = new JPasswordField(20);

rememberMeCheckbox = new JCheckBox("Remember me");

loginButton = new JButton("Login");

loginButton.addActionListener(this);

// layout

JPanel panel = new JPanel(new GridLayout(4, 2));

panel.add(usernameLabel);

panel.add(usernameField);

panel.add(passwordLabel);

panel.add(passwordField);

panel.add(new JLabel());

panel.add(rememberMeCheckbox);

panel.add(new JLabel());

panel.add(loginButton);

add(panel);

setVisible(true);

}

public static void main(String[] args) {

new aLoginFormSwing();

}

public void actionPerformed(ActionEvent e) {

String username = usernameField.getText();

String password = new String(passwordField.getPassword());

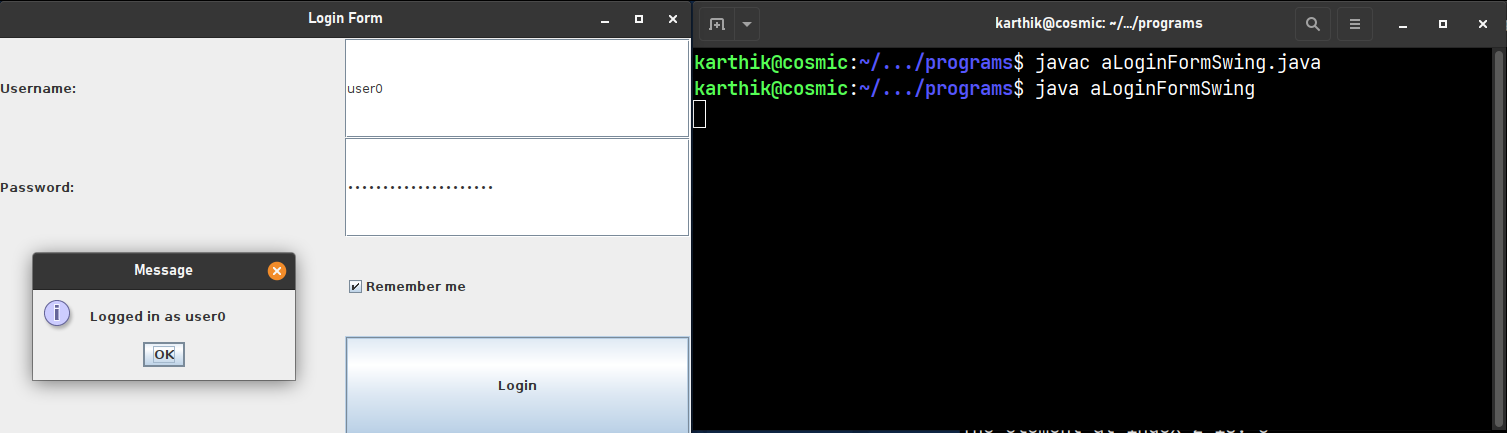
boolean rememberMe = rememberMeCheckbox.isSelected();

JOptionPane.showMessageDialog(this, "Logged in as " + username);

}

}

**Output:**



**31. Write a Java program to show all layout managers. (4 layout managers)**

**Code:**

import javax.swing.\*;

import java.awt.\*;

public class bLayoutManagers extends JFrame {

public bLayoutManagers() {

// frame

setTitle("Layout Managers");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

// components

JLabel label1 = new JLabel("Label 1");

JLabel label2 = new JLabel("Label 2");

JLabel label3 = new JLabel("Label 3");

JLabel label4 = new JLabel("Label 4");

JTextField textField = new JTextField(20);

JButton button1 = new JButton("Button 1");

JButton button2 = new JButton("Button 2");

JButton button3 = new JButton("Button 3");

JButton button4 = new JButton("Button 4");

// layouts

setLayout(new GridLayout(2, 2));

JPanel panel1 = new JPanel(new BorderLayout());

panel1.add(label1, BorderLayout.NORTH);

panel1.add(textField, BorderLayout.CENTER);

panel1.add(button1, BorderLayout.SOUTH);

add(panel1);

JPanel panel2 = new JPanel(new FlowLayout());

panel2.add(label2);

panel2.add(button2);

add(panel2);

JPanel panel3 = new JPanel(new GridLayout(2, 2));

panel3.add(label3);

panel3.add(button3);

panel3.add(new JLabel());

panel3.add(new JLabel());

add(panel3);

JPanel panel4 = new JPanel(new GridBagLayout());

GridBagConstraints c = new GridBagConstraints();

c.gridx = 0;

c.gridy = 0;

panel4.add(label4, c);

c.gridy = 1;

panel4.add(button4, c);

add(panel4);

pack();

setVisible(true);

}

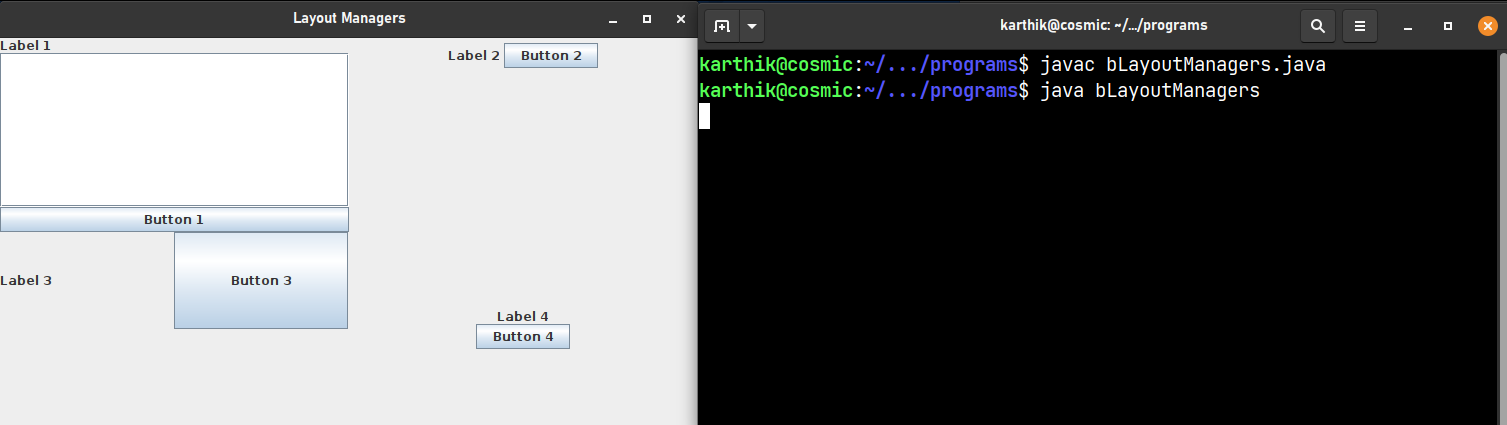
public static void main(String[] args) {

new bLayoutManagers();

}

}

**Output:**



**32. Create an applet with two buttons named ‘audio’ and ‘image’. When user will press button ‘audio’ then an audio file should play in applet, and if user press button ‘image’ then an image should see in applet window**

**Code:**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import java.io.File;

import javax.sound.sampled.\*;

import javax.imageio.ImageIO;

public class cImageAudio extends JFrame implements ActionListener {

private JButton audioButton;

private JButton imageButton;

public cImageAudio() {

// frame

setTitle("Image and Audio");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

// components

audioButton = new JButton("Audio");

imageButton = new JButton("Image");

// action listeners

audioButton.addActionListener(this);

imageButton.addActionListener(this);

// layout

setLayout(new FlowLayout());

add(audioButton);

add(imageButton);

pack();

setVisible(true);

}

public static void main(String[] args) {

new cImageAudio();

}

public void actionPerformed(ActionEvent e) {

if (e.getSource() == audioButton) {

// try play the audio file else

try {

File audioFile = new File("res/temp.wav");

AudioInputStream audioStream = AudioSystem.getAudioInputStream(audioFile);

Clip clip = AudioSystem.getClip();

clip.open(audioStream);

clip.start();

} catch (Exception ex) {

ex.printStackTrace();

}

} else if (e.getSource() == imageButton) {

// display the image

try {

File imageFile = new File("res/linux.jpg");

Image image = ImageIO.read(imageFile);

ImageIcon icon = new ImageIcon(image);

JLabel label = new JLabel(icon);

JOptionPane.showMessageDialog(this, label);

} catch (Exception ex) {

ex.printStackTrace();

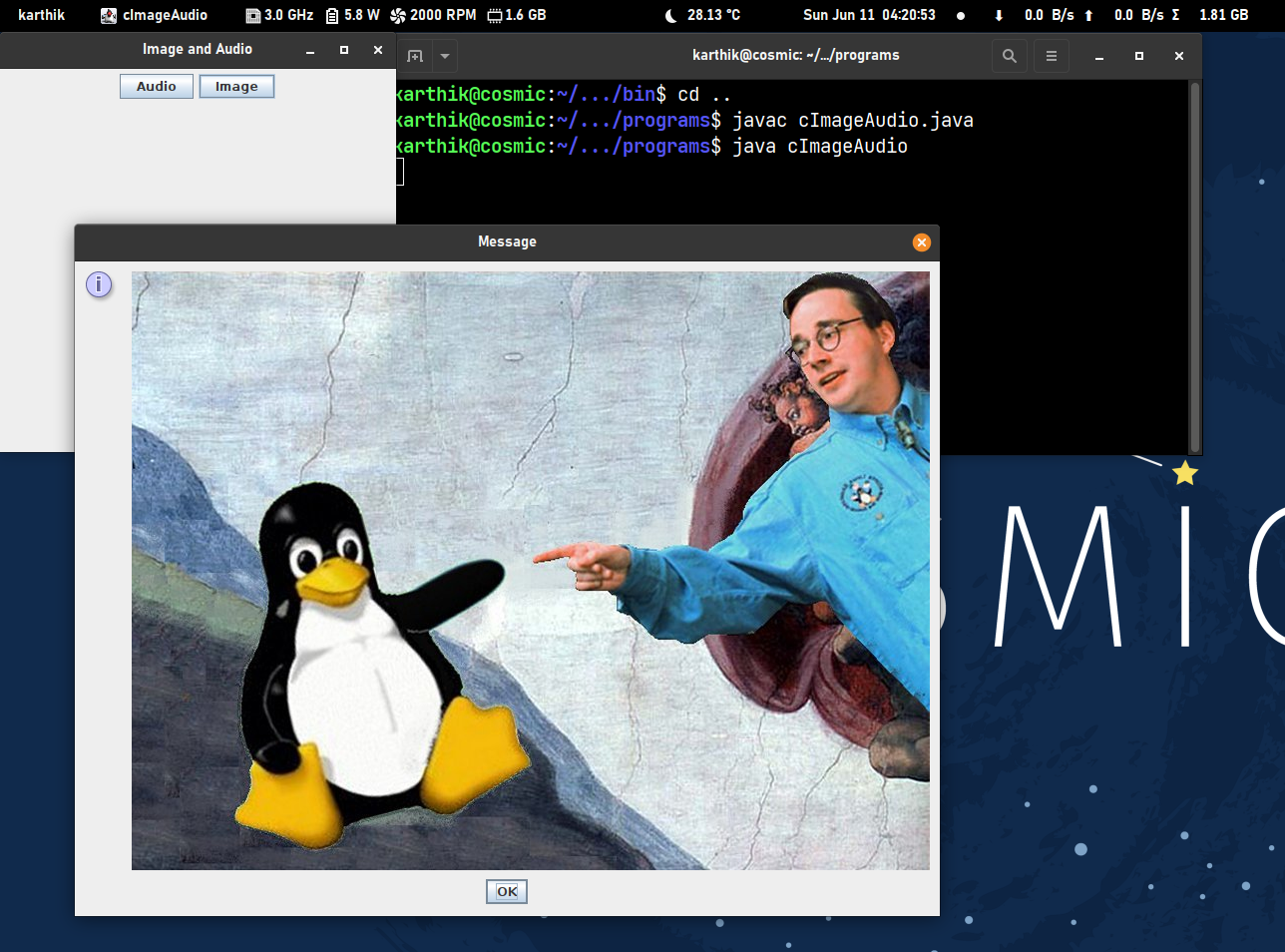
}

}

}

}

**Output:**

****

**33. Create a Java applet with three buttons ‘red’,’green’,’blue’. Whenever user presses any button the corresponding color should be seen as background color in an applet window.**

**Code:**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class dColorBG extends JFrame implements ActionListener {

private JButton redButton;

private JButton greenButton;

private JButton blueButton;

public dColorBG() {

// frame

setTitle("Background Color");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

// components

redButton = new JButton("Red");

greenButton = new JButton("Green");

blueButton = new JButton("Blue");

// action listeners

redButton.addActionListener(this);

greenButton.addActionListener(this);

blueButton.addActionListener(this);

// layout

setLayout(new FlowLayout());

add(redButton);

add(greenButton);

add(blueButton);

// frame

pack();

setVisible(true);

}

public static void main(String[] args) {

new dColorBG();

}

public void actionPerformed(ActionEvent e) {

if (e.getSource() == redButton) {

getContentPane().setBackground(Color.RED);

} else if (e.getSource() == greenButton) {

getContentPane().setBackground(Color.GREEN);

} else if (e.getSource() == blueButton) {

getContentPane().setBackground(Color.BLUE);

}

}

}

**Output:**



**34. Write a Java program in Java to implement the concept of ‘synchronization’ using thread.**

**Code:**

class Incrementer {

private int count = 0;

public synchronized void increment() {

count++;

System.out.println("Count is now " + count);

}

}

class MyThread implements Runnable {

private Incrementer Incrementer;

private String name;

public MyThread(Incrementer Incrementer, String name) {

this.Incrementer = Incrementer;

this.name = name;

}

public void run() {

for (int i = 0; i < 5; i++) {

Incrementer.increment();

try {

Thread.sleep(1000);

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

}

public class eSynchronizationUsingThread {

public static void main(String[] args) {

Incrementer Incrementer = new Incrementer();

Thread thread1 = new Thread(new MyThread(Incrementer, "Thread 1"));

Thread thread2 = new Thread(new MyThread(Incrementer, "Thread 2"));

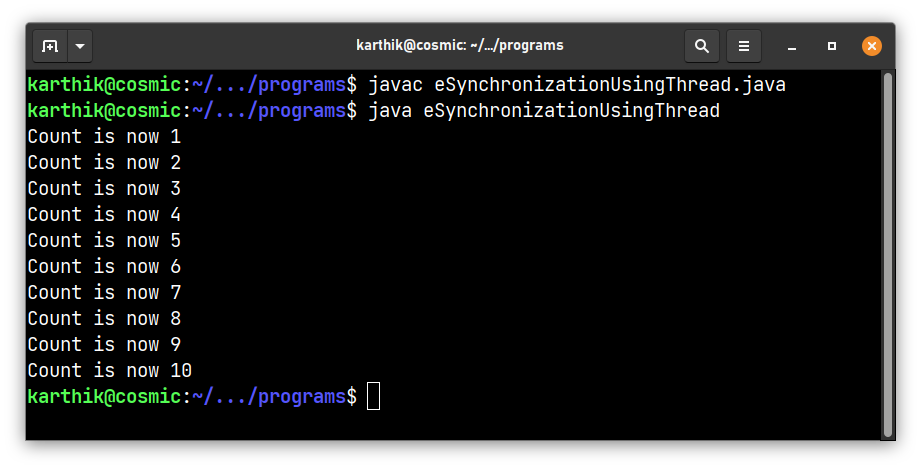
thread1.start();

thread2.start();

}

}

**Output:**



**35. Create a simple JDBC program that creates a table, stores data into it, retrieves & prints the data.**

**Code:**

import java.sql.\*;

public class fSimpleJDBCProgram {

public static void main(String[] args) {

Connection conn = null;

Statement stmt = null;

ResultSet rs = null;

try {

conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/", "root", "12345678");

stmt = conn.createStatement();

String sql = "CREATE DATABASE IF NOT EXISTS javatemp";

stmt.executeUpdate(sql);

sql = "USE javatemp";

stmt.executeUpdate(sql);

sql = "CREATE TABLE if not exists shopping\_list (id INT NOT NULL AUTO\_INCREMENT, item VARCHAR(255), PRIMARY KEY (id), quantity INT, price DOUBLE)";

stmt.executeUpdate(sql);

sql = "INSERT INTO shopping\_list (item, quantity, price) VALUES ('Apple', 5, 10.0)";

stmt.executeUpdate(sql);

sql = "INSERT INTO shopping\_list (item, quantity, price) VALUES ('Orange', 10, 5.0)";

stmt.executeUpdate(sql);

sql = "INSERT INTO shopping\_list (item, quantity, price) VALUES ('Banana', 15, 2.0)";

stmt.executeUpdate(sql);

sql = "SELECT \* FROM shopping\_list";

rs = stmt.executeQuery(sql);

while (rs.next()) {

System.out.println(rs.getString("item") + " " + rs.getInt("quantity") + " " + rs.getDouble("price"));

}

} catch (SQLException e) {

e.printStackTrace();

} finally {

try {

if (rs != null) {

rs.close();

}

if (stmt != null) {

stmt.close();

}

if (conn != null) {

conn.close();

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

}

**Output:**



**36. Write a Java program in Java to create database table using Java**

**Code:**

import java.sql.\*;

public class gCreateDatabaseTable {

public static void main(String[] args) {

Connection conn = null;

Statement stmt = null;

ResultSet rs = null;

try {

conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/", "root", "12345678");

stmt = conn.createStatement();

String sql = "CREATE DATABASE IF NOT EXISTS TestDB";

stmt.executeUpdate(sql);

System.out.println("Database TestDB created successfully...");

sql = "USE TestDB";

stmt.executeUpdate(sql);

sql = "CREATE TABLE if not exists student (id INT NOT NULL, name VARCHAR(255), enrol\_num INT, status INT, PRIMARY KEY (id))";

stmt.executeUpdate(sql);

System.out.println("Table student created successfully...");

} catch (SQLException e) {

e.printStackTrace();

} finally {

try {

if (rs != null) {

rs.close();

}

if (stmt != null) {

stmt.close();

}

if (conn != null) {

conn.close();

}

} catch (SQLException e) {

e.printStackTrace();

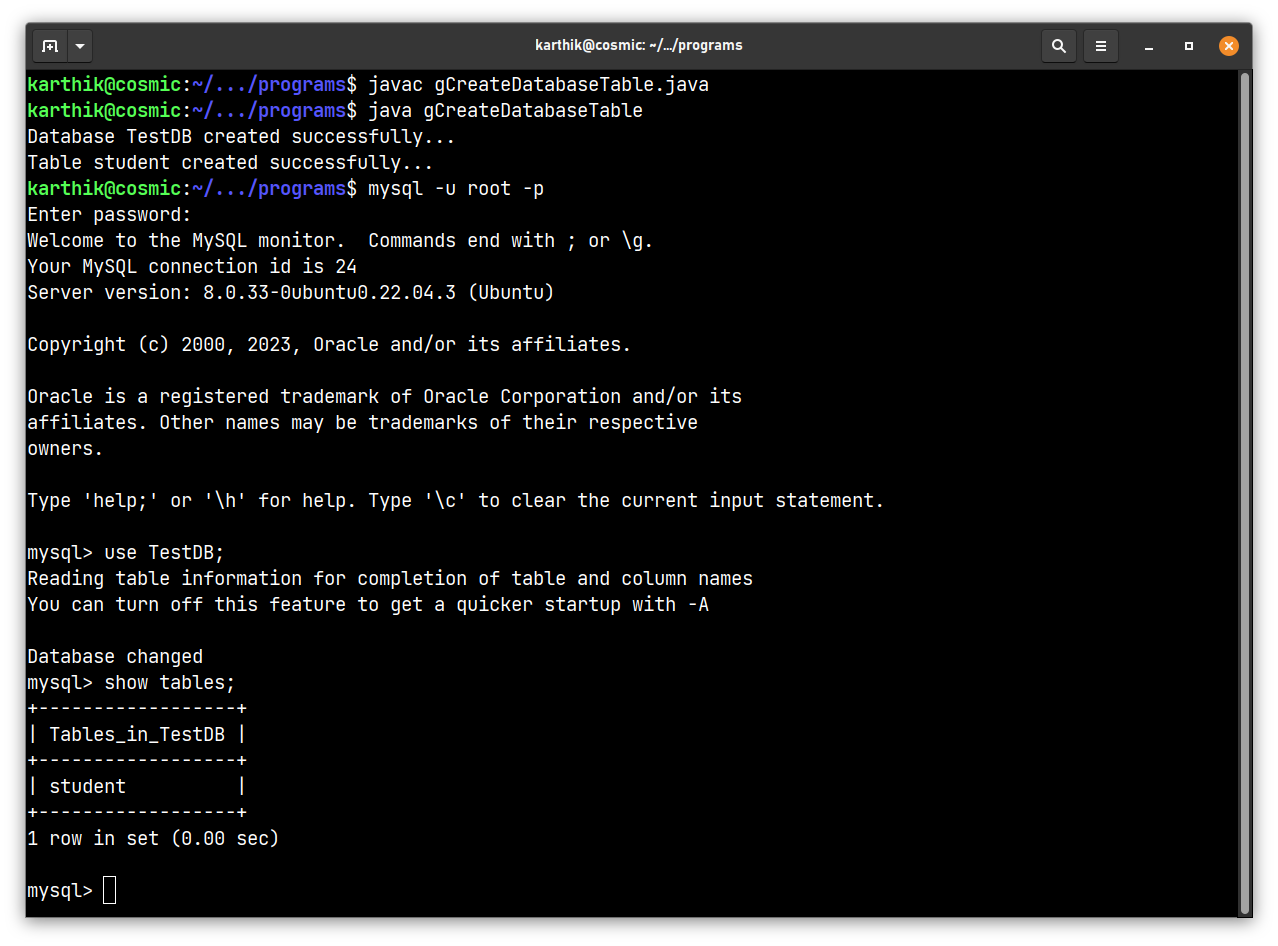
}

}

}

}

**Output:**



**37. Write a Java program in Java to insert, update, delete & select records**

**Code:**

import java.sql.\*;

public class hInsertUpdateDeleteSelect {

public static void main(String[] args) {

Connection conn = null;

PreparedStatement pstmt = null;

ResultSet rs = null;

try {

conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/TestDB", "root", "12345678");

String insertSql = "INSERT INTO student (id, name, enrol\_num, status) VALUES (?, ?, ?, ?)";

pstmt = conn.prepareStatement(insertSql);

pstmt.setInt(1, 1);

pstmt.setString(2, "Ramesh Kumar");

pstmt.setInt(3, 12345);

pstmt.setInt(4, 1);

pstmt.executeUpdate();

System.out.println("Record inserted successfully.");

pstmt.setInt(1, 2);

pstmt.setString(2, "Sandra");

pstmt.setInt(3, 23456);

pstmt.setInt(4, 1);

pstmt.executeUpdate();

System.out.println("Record inserted successfully.");

String updateSql = "UPDATE student SET name = ?, enrol\_num = ?, status = ? WHERE id = ?";

pstmt = conn.prepareStatement(updateSql);

pstmt.setString(1, "Jane");

pstmt.setInt(2, 54321);

pstmt.setInt(3, 0);

pstmt.setInt(4, 1);

pstmt.executeUpdate();

System.out.println("Record updated successfully.");

String deleteSql = "DELETE FROM student WHERE id = ?";

pstmt = conn.prepareStatement(deleteSql);

pstmt.setInt(1, 1);

pstmt.executeUpdate();

System.out.println("Record deleted successfully.");

String selectSql = "SELECT \* FROM student";

pstmt = conn.prepareStatement(selectSql);

rs = pstmt.executeQuery();

while (rs.next()) {

int id = rs.getInt("id");

String name = rs.getString("name");

int enrolNum = rs.getInt("enrol\_num");

int status = rs.getInt("status");

System.out.println("id: " + id + ", name: " + name + ", enrol\_num: " + enrolNum + ", status: " + status);

}

} catch (SQLException e) {

e.printStackTrace();

} finally {

try {

if (rs != null) {

rs.close();

}

if (pstmt != null) {

pstmt.close();

}

if (conn != null) {

conn.close();

}

} catch (SQLException e) {

e.printStackTrace();

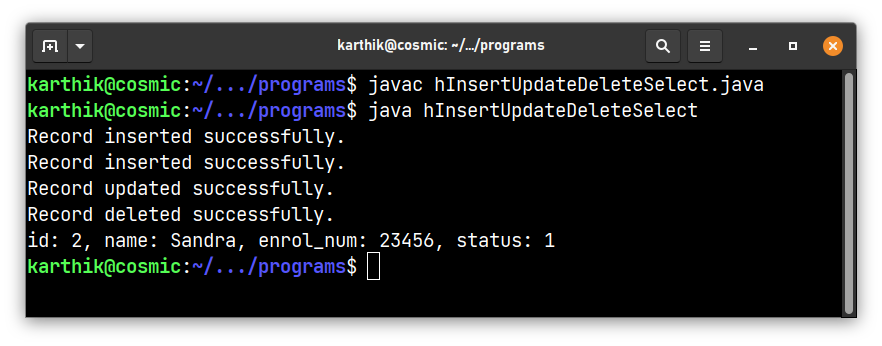
}

}

}

}

**Output:**



**38. Write Java program to read input from java console.**

**Code:**

import java.util.Scanner;

public class iInputFromJavaConsole {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter your name: ");

String name = scanner.nextLine();

System.out.print("Enter your age: ");

int age = scanner.nextInt();

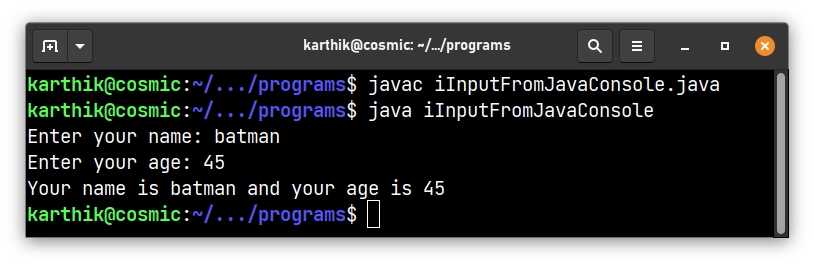
System.out.println("Your name is " + name + " and your age is " + age);

scanner.close();

}

}

**Output:**



**39. Write a Java program to implement file handling(both reading & writing to a file)**

**Code:**

import java.io.\*;

public class jFileHandling {

public static void main(String[] args) {

try {

File file = new File("res/temp.txt");

FileWriter writer = new FileWriter(file);

writer.write("Hello World! This line was added to the file");

writer.close();

System.out.println("Data written into file successfully.");

System.out.println("Reading file...");

FileReader reader = new FileReader(file);

BufferedReader bufferedReader = new BufferedReader(reader);

String line;

while ((line = bufferedReader.readLine()) != null) {

System.out.println(line);

}

bufferedReader.close();

} catch (IOException e) {

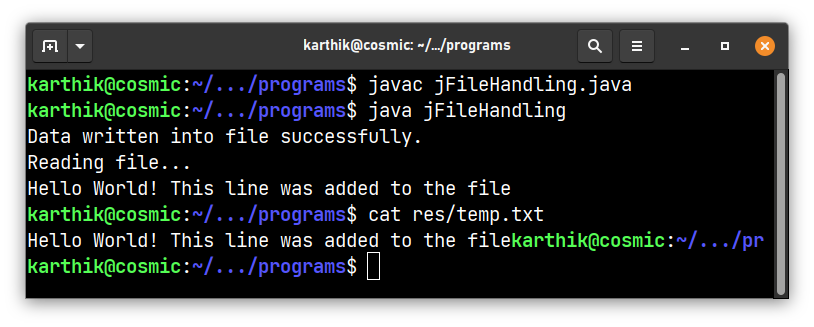
e.printStackTrace();

}

}

}

**Output:**



**40. Write a Java program on anonymous classes**

**Code:**

public class kAnonymousClasses {

public static void main(String[] args) {

Runnable runnable = new Runnable() {

public void run() {

System.out.println("Hello from an anonymous class!");

}

};

Thread thread = new Thread(runnable);

thread.start();

}

}

**Output:**

